Enhancing Wireless Internet Links

George Xylomenos and George C. Polyzos
{xgeorge,polyzos}@cs.ucsd.edu
University of California, San Diego,
Department of Computer Science & Engineering,
La Jolla, California, 92093-0114, U.S.A.

ABSTRACT

We describe a novel link layer architecture to enhance the performance of Internet protocols over wireless links. The degraded performance of these links on the Internet and the inadequacy of existing approaches of overcoming these problems in a protocol independent manner, motivate our solution. Our link layer provides multiple services and performance feedback to higher layers, thus supporting adaptive protocols and applications. In addition, our approach can serve as the basis for future Internet evolution towards Quality of Service provision. We also describe our ongoing research and implementation directions.

I. INTRODUCTION

During the past few years, wireless and portable communications have been expanding rapidly. As cellular telephony is evolving worldwide towards fully digital systems, the opportunity arises to extend the reach of the advanced digital data services of the future to mobile users. At the same time, Wireless Local Area Networks (WLANs) are becoming available that can transparently link wireless hosts to the Internet. Despite their differences, cellular systems and WLANs share common characteristics, setting them apart from existing wired and satellite wireless networks. In contrast to satellite links, cellular and WLAN links exhibit low propagation delays, while in contrast to wired links, they suffer from bandwidth shortages and higher loss rates, due to the need to share the RF spectrum and the imperfections associated with this process.

As these systems gain in popularity, there is increasing pressure to integrate them with the Internet as seamlessly as possible. Superficially, this is an easy task as Internet protocols are designed to accommodate diverse network technologies, making only minimal assumptions about their capabilities. In practice however, the design of these protocols has been influenced by assumptions that hold for wired, but not for wireless networks. As an example, a WLAN that suffers only a 2% IP packet loss, causes TCP throughout to drop to only 47% of its value in the absence of losses, for local area transfers [2]. The corresponding figure for wide area transfers is only 23%. Even worse, a cellular system that suffers a 2% loss over its short, speech optimized, frames, would suffer IP packet loss of 64%, thus reducing TCP throughput to nearly zero [6].

We believe that this presents to the networking community both a challenge and an opportunity. The challenge is to overcome the limitations of existing protocols when employed over wireless media. The opportunity is to prohibit such media dependencies from causing trouble in the future. We propose enhancements to the link layer that accomplish this goal and at the same time ease the evolution of the Internet towards Quality of Service (QoS) support. In Section II we examine existing approaches to these problems and point out their shortcomings. Section III discusses the evolution of the Internet and wireless systems in general and the resulting requirements for network protocols. In Section IV we outline our link layer approach, while in Section V we describe the services that will be offered by our protocols and their service interface. Section VI discusses some preliminary implementation ideas for such protocols. We conclude in Section VII.

II. RELATED WORK

Studies of the performance of TCP over wireless links have shown that the cause of its abysmal performance over wireless links is its design assumption that losses are due to congestion [4]. Congestion avoidance is thus triggered repeatedly even for low error rates, in turn diminishing the effective throughput of TCP and wasting available bandwidth over the wireless link, which may well be the bottleneck of the path. In addition, recovery is performed with end-to-end retransmissions, considerably increasing the delay visible to the user.

One school of thought has concentrated on avoiding such problems by modifications at the transport layer. The main idea is to split an end-to-end path that contains wireless links to segments that consist of wired or wireless links only. Separate transport connections are then established on each of these segments, communicating through a relay agent that copies data from one to the other [1]. Over the wireless parts of the path, protocols better tuned to recovery from wireless losses can be employed instead of TCP. With this scheme recovery is isolated over the problematic links only and is faster since it is not performed end-to-end. Deployment of such protocol modifications can be localized to wireless links only, effectively hiding their local problems.

Splitting transport connections on the other hand violates their end-to-end semantics and mandates an additional protocol layer for this purpose. It is questionable how well such an approach would perform over paths with multiple wireless links, although it is clear that in the limit it would degenerate to a hop-by-hop scheme, requiring modifications at numerous pivot points. More importantly, this method violates protocol layering by closely coupling the transport layer design to link

characteristics. As a result, it is only appropriate for a particular protocol and underlying problem and will not scale properly as new problems become apparent in the future.

Another school of thought has advocated local modifications to the link layer. With this scheme unacknowledged TCP data are buffered at each end of the wireless link. Losses are detected by timeouts or duplicate acknowledgments. Local retransmissions are used for recovery and duplicate acknowledgments are dropped to avoid triggering congestion recovery at the transport layer [3]. By exploiting existing TCP messages this scheme avoids additional control overhead and integrates transparently with TCP. It also outperforms the transport layer schemes discussed above, while preserving end-to-end semantics [2].

This approach also violates protocol layering by using transport layer information and messages for link layer purposes. An interesting observation is that it performs better in the direction from the wired network to a wireless end host. In the reverse direction TCP acknowledgments arrive too late to be useful for local recovery. A similar issue arises when the wireless link is not the last hop of an end-to-end path. At the same time, this is another solution that only solves a specific problem for a specific protocol, thus its scalability is limited. Moreover, its dependence on TCP mechanisms requires the protocol to be tuned for the particular TCP variant in use for maximum effectiveness.

Finally, numerous Radio Link Protocols (RLPs) have been proposed for cellular data services. Traditional approaches offer complete recovery at the link layer at the expense of greatly varying delays [8]. These delays however can cause TCP to timeout and trigger congestion recovery end-to-end. Another approach tries to avoid such adverse interactions by only offering limited recovery, leaving complete recovery to higher layers [6]. Although this scheme works well with TCP, it suffers from the same problem as the traditional approaches: it is inflexible since it only offers a single type of service.

III. INTERNET & WIRELESS NETWORK EVOLUTION

Due to the increased user populations and the demand for more bandwidth for data services, wireless networks seem to be evolving towards hierarchical cellular systems. At the lowest level of the hierarchy, picocellular systems (very small cell diameters) will allocate high bandwidths to a few local users. At the highest level, satellite systems will provide ubiquitous coverage, with each satellite supporting many users in a (large diameter) macrocell. Thus each successive layer in the hierarchy will overlay the ones below it, offering less bandwidth but wider coverage [7]. Users would choose the network to use based on application requirements and service costs.

Apart from increasing the frequency of horizontal handoffs (between cells in the same system) while a user is roaming in a picocellular network, hierarchical systems impose the requirement for vertical handoffs (from one hierarchy level to the next). This implies that wireless communications will have to face two levels of performance variations: short term due to the changing environment of wireless links, and medium term due to handoffs to media with different characteristics. Short term variations may be hidden from end-to-end layers by employing local recovery mechanisms that adapt to the current state of the link. Medium term variations on the other hand would require end-to-end protocols and applications to adapt their operation [5]. To enable higher layers to detect such events in a timely manner, it is preferable to utilize information from lower layers, thus avoiding confusion between congestion and medium term path modifications.

The Internet on the other hand seems to be evolving towards the provision of Quality of Service (QoS) guarantees. Given the dynamic and unpredictable nature of the Internet architecture, these guarantees will most likely be probabilistic to some extent. Since existing link layer services are best effort, it is hard to compose flexible end-to-end services that could satisfy diverse application requirements. It is also hard to predict how an end-to-end connection would perform, due to the lack of local performance information. The transition of the Internet to an integrated services packet network [5] with QoS support would be considerably eased by the provision of multiple QoS classes and link performance metrics for each class by lower layers.

IV. LINK LAYER ARCHITECTURE

Solving the problems imposed by the need to integrate wireless networks with the Internet is not a task for a single protocol layer. In order to properly allocate responsibilities to each layer we need to consider how to avoid adverse interactions with future protocols. Furthermore, we should also take into account the evolution of the Internet in general, in order to make our solutions ease its transition to its future form. Our general design guidelines are:

- Cleanly allocate tasks to each protocol layer.
- Adhere to protocol layering during design.
- Adopt an approach supporting multiple protocols.
- Cater to the requirements of a QoS based Internet.

Since the pivot point between local and end-to-end layers is the network layer, it is the most appropriate layer to deal with mobility. Since it multiplexes traffic from different flows, it could also handle packet scheduling based on QoS classes if the Internet evolves in this direction. Local tasks such as error recovery are best handled by the link layer. Medium specific characteristics for each type of wireless network are thus isolated and encapsulated locally. Events such as handoffs on the other hand are beyond its reach. Horizontal handoffs may be possible to hide within the network layer, but vertical handoffs are within the responsibility of the transport and application layers. Possible adaptive operation in multiple layers would require them to co-ordinate in order to avoid adverse interactions.

To fulfill these requirements, we intent to build a family of link layer protocols that are optimized for wireless links and support Internet protocols. Each protocol will employ mechanisms that depend on the characteristics of the underlying medium. The mechanisms will adapt to current link conditions in an attempt to hide local performance variations from higher layers as far as possible. The protocols will also support multiple QoS classes so as to support the requirements of different existing and future higher layer protocols and applications. These QoS classes are meant to satisfy generic requirements rather than fit the specific characteristics of TCP or UDP.

With this approach it is possible to optimize the performance of multiple current protocols without penalizing future ones. To ease the task of higher layers and applications that are adaptive, our protocols will also export information on link performance metrics and provide notifications to any interested parties about events that could signify medium term performance variations. Thus, service requests will flow downwards through the protocol stack, while information will flow upwards from the link layer to adaptive higher layers and applications.

To integrate our protocols into a next generation protocol stack, they will all have the same interface to higher layers, thus effectively isolating them from link specific mechanisms. Higher layers supporting QoS classes will then use the most appropriate link layer services without media specific knowledge. The exported link performance metrics will allow higher layers to estimate end-to-end path capabilities and performance, enabling them in turn to offer flexible QoS classes to their users. In co-ordination with handoff notifications, they will also allow higher layers to reconfigure their end-to-end services based on updated information when vertical handoffs occur. Even if only some links of a path offer such advanced characteristics, suitable higher layers can compose more flexible end-to-end services using service substitutions [10]. Deployment of our link layer is both easy, as it is transparent to unaware higher layers and only requires local modifications, and attractive, since it will benefit even existing higher layers via its recovery mechanisms. When in place, our link layer would ease the evolution of higher layers to support QoS.

V. LINK LAYER SERVICES & INTERFACE

Besides defining the way our link layer will interface with higher layers, we also need to define the content of these interactions. For the QoS based protocols discussed above, we need to specify the supported service classes and the exported link performance metrics. Regardless of the underlying medium and its characteristics, a link layer protocol may make tradeoffs between three metrics: throughput, delay and reliability. Since wireless link performance is influenced by transient environmental factors and thus unpredictable in general, we can only support QoS classes that prioritize between metrics rather than make guarantees about them. Based on common application and transport protocol requirements, we plan to support the following services:

1) Fully Reliable
2) Delay Sensitive
3) Throughput Sensitive
4) Mostly Reliable, Delay Sensitive
5) Mostly Reliable, Throughput Sensitive

The first three services optimize one of the three metrics without regard to the other two, while the last two provide a tradeoff between reliability and either delay or throughput. Note that reliable services may be offered by higher layers over non fully reliable link services, through additional end-to-end error recovery mechanisms. In this framework, it is reasonable to export throughput, delay and reliability metrics for each QoS class, thus enabling higher layers to estimate the performance of each class, use the one most appropriate for their needs, and (optionally) adapt their mechanisms accordingly. Initial values for these metrics are determined using physical layer information, and they are updated based on current measurements. We have assumed that the network layer will appropriately schedule packets from different QoS classes and prioritize them within each class, if needed. Our link layer then also needs to preserve the share allocated to each class and the priorities of packets within a class.

We have attempted to make the interface to our link protocol transparent to and compatible with existing higher layers. Rather than modifying the packet sending calls of the link layer, we have instead decided to determine the QoS class of each packet by using the IP Type of Service (ToS) field. This field contains a single bit flag for each of our key metrics (throughput, delay and reliability), thus appropriate combinations of these flags can be directly mapped to our proposed QoS classes. Priorities within each class can similarly be handled by the three bit priority subfield of the ToS field. Sharing the link among QoS classes is based on the relative loads from each QoS class that the network layer offers to the link layer during a time interval. Link metrics on the other hand will be exported via virtual device files (one per class), in units such as milliseconds and bytes, rather than frames or packets. This will enable interested parties to access these metrics by reading files, transforming them to their own units without knowledge of the units used by other layers. An alternative approach would for applications to explicitly define an acceptable range for a metric, and be notified when this range is exceeded [9]. We decided against this approach based on elegance, but it could easily be handled by our architecture.

The other service offered by our link layer is notification of connections and disconnections to higher layers. Since the link layer has detailed knowledge of the underlying medium, it may be able to detect such events very fast by exploiting physical layer information. Alternatively, link layer mechanisms may be used for detection, albeit at a slower speed. Since prolonged error periods may cause the link layer to disconnect and later reconnect to the same peer, these are not reliable handoff notifications. Network and higher layers can still employ these notifications to detect periods of no connectivity as well as horizontal and vertical handoffs, and appropriately modify their behavior. An unobtrusive interface to this feature is an upcall: interested parties register a function with the link layer, which is called when the event of interest occurs. The network layer could transform such upcalls to reliable handoff notifications. This asynchronous mechanism complements the synchronous mechanisms above, by indicating to higher layers when to review the exported link metrics.

VI. IMPLEMENTATION CONCERNS

The choice of mechanisms for the actual implementation of protocols that fit our architecture is a matter of ongoing research. However, some important issues have already emerged from our research, causing us to formulate some general guidelines. Although numerous error recovery schemes could be employed, it is clear that choosing the most appropriate one depends on underlying media characteristics and the requirements

of a particular QoS class. For example, the provision of reliability with low delay is possible using Automatic Repeat Request (ARQ) on wireless LANs, where delays are on the order of a few milliseconds. In contrast, on low speed medium distance cellular links, each retransmission could cause delays on the order of several tens of milliseconds, thus making Forward Error Correction (FEC) attractive. Similarly, a mechanism should adapt fast enough to link performance variations [9], where the feasible speed depends on medium constraints while the desired speed depends on QoS constraints. It is often unclear how medium characteristics such as error rates and distributions influence particular mechanisms in practice. For this reason, we expect protocol development to be followed by a considerable period of testing and tuning.

The support of multiple services can considerably complicate the link layer protocol itself. To avoid performance degradations or unrealistic hardware requirements due to protocol overhead, we should strive for simplicity and efficiency. To the extent possible protocol mechanisms should be shared between different QoS classes. The very nature of multiple service support may mandate the use of mechanisms that are suboptimal for any given QoS class, but are more efficient overall when a mixed class traffic is considered.

To support protocol deployment, we need to make them attractive by providing enhanced performance for existing higher layers. Rather than customizing mechanisms around TCP or UDP, we instead accommodate them within our generic services. TCP fits the Mostly Reliable, Delay Sensitive service. Since UDP is used for diverse applications, different QoS classes are more appropriate for each one. This raises the question of how to communicate these application requirements to the link layer. As we have discussed, we intend to use the IP ToS field for this purpose, so our implementation will provide applications with calls that explicitly set these fields. When application sources cannot be modified, a temporary workaround will be required. A classifier module would be inserted between the network and link layers, mapping IP packets to link services depending on the transport protocol and ports employed. Since this is only a heuristic method of deriving application requirements, it should be conservative and only err on the safe side, to avoid performance degradations. These modules will be made redundant when use of the ToS field becomes more prevalent. Note that the classifier is a transparent performance optimization option and is not required for compatibility.

Our implementation effort is based on a testbed that includes both desktop and portable Pentium PCs running the Linux OS, and are interconnected by a Digital RoamAbout wireless LAN. IS-95 cellular data links will be added when the service becomes available. For the operating system in use, we will use devices on a virtual file system to export link metrics, signals to deliver notifications (upcalls) to registered parties, and socket options to set ToS values. Our measurements using the raw services of the wireless LAN have uncovered some interesting effects that will be documented separately, and indicate that considerable performance improvements may be achieved by a more appropriate link layer. Protocol testing will also employ a partially developed in kernel error simulator, that will accurately reproduce various operating conditions under our control. Calibration of the error model to use is also an ongoing effort. This approach will allow us to test various protocol mechanisms and parameters under the same assumptions, within a real operational testbed, thus minimizing the inaccuracies normally associated with simulations.

VII. CONCLUSION

We have described the problems that arise when cellular and wireless LAN networks are employed on the Internet, and how they may be aggravated in the future. We examined a number of approaches for overcoming wireless link limitations, and pointed out their inability to support multiple protocols and future extensions. We then proposed a novel link layer architecture that could not only enhance the performance of existing protocols, but also ease the transition into a future Internet incorporating QoS provisions. We also described the services that this link layer would support and its content rich interface with higher layers, and concluded with some issues related to our ongoing research and implementation effort.

REFERENCES

[1] B.R. Badrinath, A. Bakre, T. Imielinski, and R. Marantz, “Handling mobile clients: A case for indirect interaction,” In Proceedings of the 4th Workshop on Workstation Operating Systems, 1993, pp. 91–97.

[2] H. Balakrishnan, V.N. Padmanabhan, S. Seshan, and R.H. Katz, “A comparison of mechanisms for improving TCP performance over wireless links,” In Proceedings of the ACM SIGCOMM, 1996, pp. 256–267.

[3] H. Balakrishnan, S. Seshan, and R.H. Katz, “Improving reliable transport and handoff performance in cellular wireless networks,” Wireless Networks, vol. 1, 1995, pp. 469–481.

[4] R. Cáceres and L. Iftode, “Improving the performance of reliable transport protocols in mobile computing environments,” IEEE Journal on Selected Areas in Communications, vol. 13, 1995, pp. 850–857.

[5] D.D. Clark, S. Shenker, and L. Zhang, “Supporting real-time applications in an integrated services packet network: Architecture and mechanism,” In Proceedings of the ACM SIGCOMM, 1992, pp. 14–26.

[6] P. Karn, “The Qualcomm CDMA digital cellular system,” In Proceedings of the USENIX Mobile and Location-Independent Computing Symposium, 1993, pp. 35–39.

[7] R.H. Katz and E.A. Brewer, “The case for wireless overlay networks,” Proceedings of the SPIE–The International Society for Optical Engineering, 2267, 1996, pp. 77–88.

[8] S. Nanda, R. Ejzak, and B.T. Doshi, “A retransmission scheme for circuit-mode data on wireless links,” IEEE Journal on Selected Areas in Communications, vol. 12, 1994, pp. 1338–1352.

[9] B. Noble, M. Satyanarayanan, D. Narayanan, J.E. Tilton, J. Flinn, and K.R. Walker, “Agile application-aware adaptation for mobility,” In Proceedings of the 16th ACM Symposium On Operating Systems Principles, 1997.

[10] S. Shenker and L. Breslau, “Two issues in reservation establishment,” In Proceedings of the ACM SIGCOMM, 1995, pp. 14–26.